Norwegian University of Science and Technology

Operating Systems

Lecture 1: Motivation and History

Michael Engel

whoami?

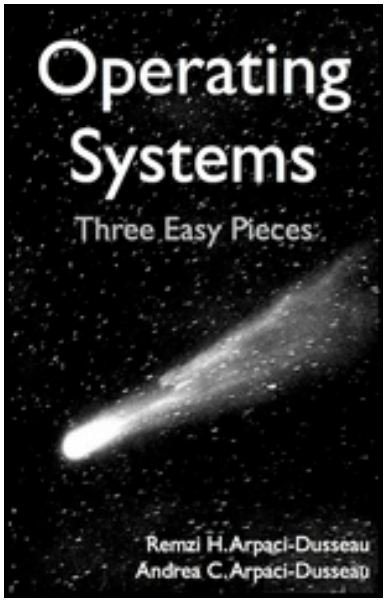
- Michael Engel (michael.engel@ntnu.no, http://folk.ntnu.no/michaeng/)
- Studied computer engineering and applied mathematics (Univ. Siegen)
- PhD (Univ. Marburg) 2005
- Assist. Prof. TU Dortmund 2007–14
- Leeds Beckett U., Oracle Labs UK 2014–16
- Assoc. Prof. Coburg Univ. 2016–19
- Assoc. Prof. NTNU 2020–…

- Research Interests

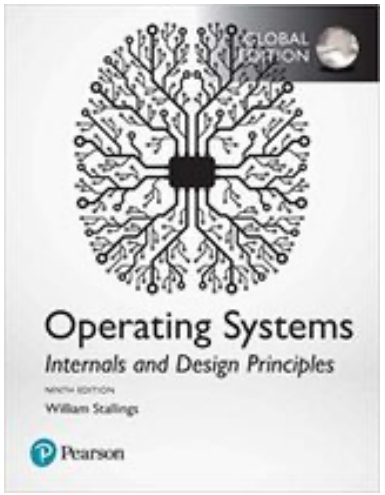
  Compilers, operating systems, parallelization, dependability, embedded systems

Literature

Authors	Remzi H. Arpaci-Dusseau and Andrea C. Arpaci-Dusseau
Title	Operating Systems: Three Easy Pieces
Available	Free PDF download: http://pages.cs.wisc.edu/~remzi/OSTEP/



Author	William Stallings
Title	Operating Systems - Internals and Design Principles, 9th Global Edition
ISBN	9781292214290



+ additional papers, articles, … on my web page: http://folk.ntnu.no/michaeng/tdt4186_21/

Overview

- Learning objectives
- Definition of an operating system
- History: the evolution of computers and operating systems
- From batch processing over multiprogramming to interactive computer use
- Semester overview

Learning objectives

- Acquire basic knowledge about **operating systems**
  - Functionality and structure
  - Algorithms and implementation
  - Examples of implementation details in Linux
- First experiences with **system programming**
  - Exercises in C running under Unix
- Understanding what is going on inside a **computer system**
- Learn about current **trends** and **challenges**
  - At least some important ones…

Definitions: what is an operating system?

- “Operating system: software that controls the operation of a computer and directs the processing of programs (as by assigning storage space in memory and controlling input and output functions)” [Merriam-Webster]
- “The operating system is software that manages every part of a computer system—all hardware and all other software. To be specific, it controls every file, every device, every section of main memory, every nanosecond of processing time, and every network connection. It controls who can use the system and how. In short, it is the boss—without it, nothing can happen” [encyclopedia.com]

Definitions: what is an operating system?

- “It is hard to pin down what an operating system is other than saying it is the software that runs in kernel mode—and even that is not always true. Part of the problem is that operating systems perform two basically unrelated functions:
  - providing application programmers (and application programs, naturally) a clean abstract set of resources instead of the messy hardware ones
  - and managing these hardware resources. Depending on who is doing the talking, you might hear mostly about one function or the other.”
    [Tanenbaum, Modern Operating Systems]

Definitions: what is an operating system?

- “An OS is a program that controls the execution of application programs, and acts as an interface between applications and the computer hardware. It can be thought of as having three objectives:
  - Convenience: An OS makes a computer more convenient to use.
  - Efficiency: An OS allows the computer system resources to be used in an efficient manner.
  - Ability to evolve: An OS should be constructed in such a way as to permit the effective development, testing, and introduction of new system functions without interfering with service.”
    [Stallings, Operating Systems]

Definitions: what is an operating system?

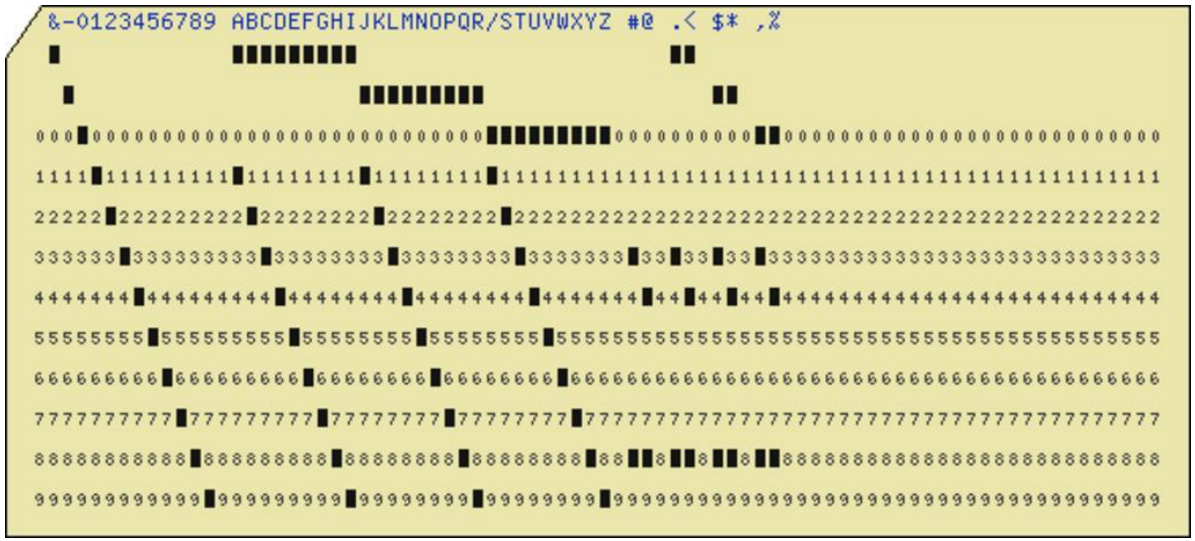
- “There is a body of software, in fact, that is responsible for making it easy to run programs (even allowing you to seemingly run many at the same time), allowing programs to share memory, enabling programs to interact with devices, and other fun stuff like that. That body of software is called the operating system (OS), as it is in charge of making sure the system operates correctly and efficiently in an easy-to-use manner.”
  [Arpaci-Dusseau, Operating Systems - Three Easy Pieces]

Definitions summary

- There are many definitions of the term “operating system”
- Common ideas are:
  - The OS serves the users and their programs, it is never an end to itself
  - The OS has to know the hardware in detail and provides suitable abstractions to application programs
- Hardware and application requirements determine the services provided by an OS
  - From this, its structure and functionality are derived
- To understand which hardware abstractions are provided by operating systems today, we take a look at their **history of development** along with advances in hardware and typical applications

In the beginning…

- Punched cards – paper cards with holes indicating a “1”
- Since 1725 used to control weaving looms (Jacquard)
- Used by Hermann Hollerith for the 1890 US census
  - Hollerith and two other companies formed IBM later
- Used until the 1970s as versatile memory



[From Wikipedia by Arnold Reinhold, CC0 1.0]

The first Norwegian computer

- NUSSE – “Norsk Universell Siffermaskin Selvstyrt Elektronisk” [1] – was developed at UiO, Sentralinstitutt for industriell forskning, by Thomas Hysing, Ole Amble and Tor Evjen
- Constructed between 1950 and 1955
- 1000 vacuum tubes
- 2 kB main memory
- Clock time 0.024 ms
  - ca. 40 kHz!
- ca. 100 arithmetic operations/second
- Now exhibited at the Norsk teknisk museum in Oslo [2]

[From Wikipedia by Mahlum, Public Domain]

The first computer at NTNU (NTH)

- …came from Denmark in 1962 to NTH: GIER "Geodætisk Instituts Elektroniske Regnemaskine" [3,4]
- Already transistorized, ca. 40 built
- 1024 words of 42-bit main memory (~5 kB)
- Fixed point addition: 49 µs, multiplication: 180 µs
  Floating point addition: 93 µs, multiplication: 170 µs
- Magnetic drum, 960 tracks with 40 words each (~200 kB)
- Already provided:
  - operating system
  - Algol-60 compiler
  - runtime library with virtual memory management
- GIER simulator [5], video [6]

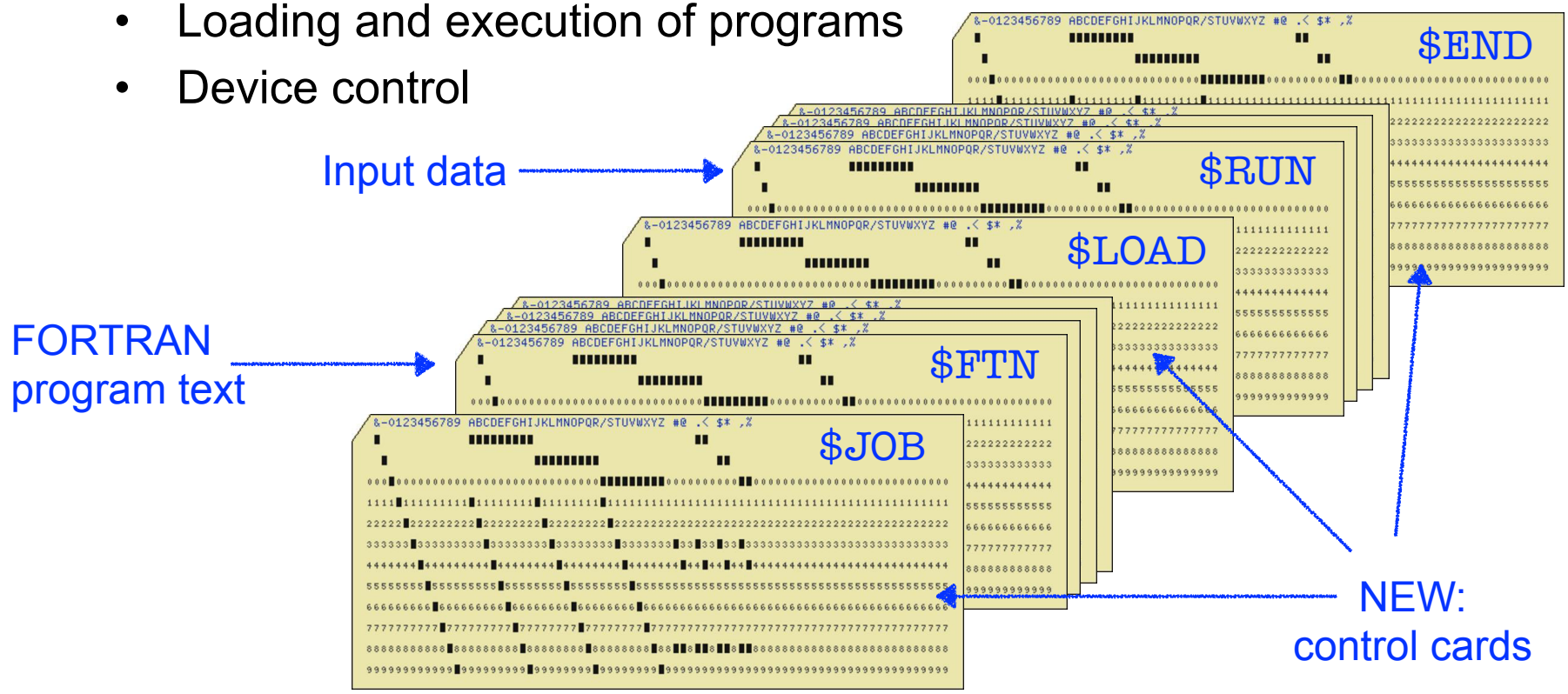
[From Wikipedia by Mahlum, Public Domain]

Serial processing

- Non-interactive programming of computers
  - Commonly in machine language
  - Input via punched card reader
  - Output via printer
  - Errors indicated using lamps
- Compute time scheduled with a paper calendar
  - Waste of compute time due to overallocation or termination of programs due to errors
- Minimal CPU utilisation
  - Most of the time was spent waiting for **slow I/O devices** (punched card reader, printer)
- First system software as reusable **program libraries**
  - Linker, loader, *debugger*, device drivers, …

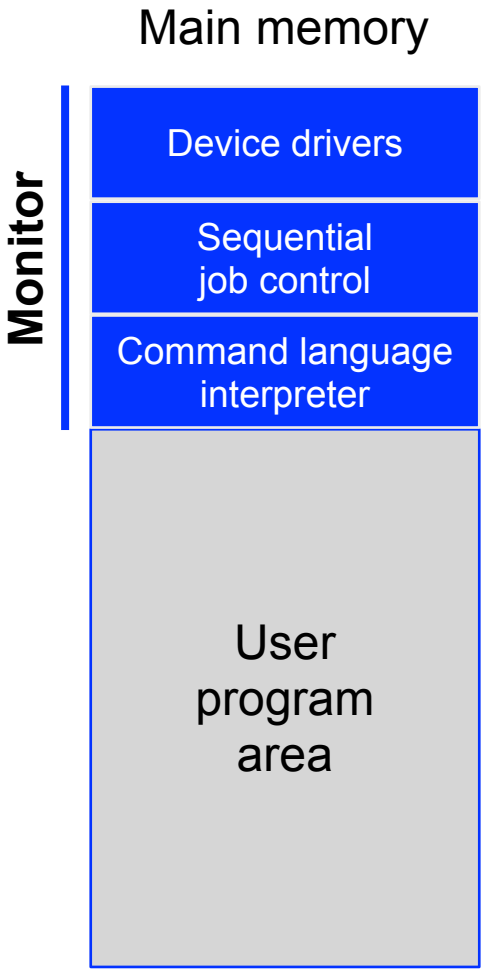
Simple batch systems (from 1955)

- Reduced frequency of operator interactions
- First operating systems: “resident monitors”
  - Interpretation of *job* control commands
  - Loading and execution of programs
  - Device control



Simple batch systems (from 1955)

The **monitor** stayed **resident** in memory and executed one application after the other



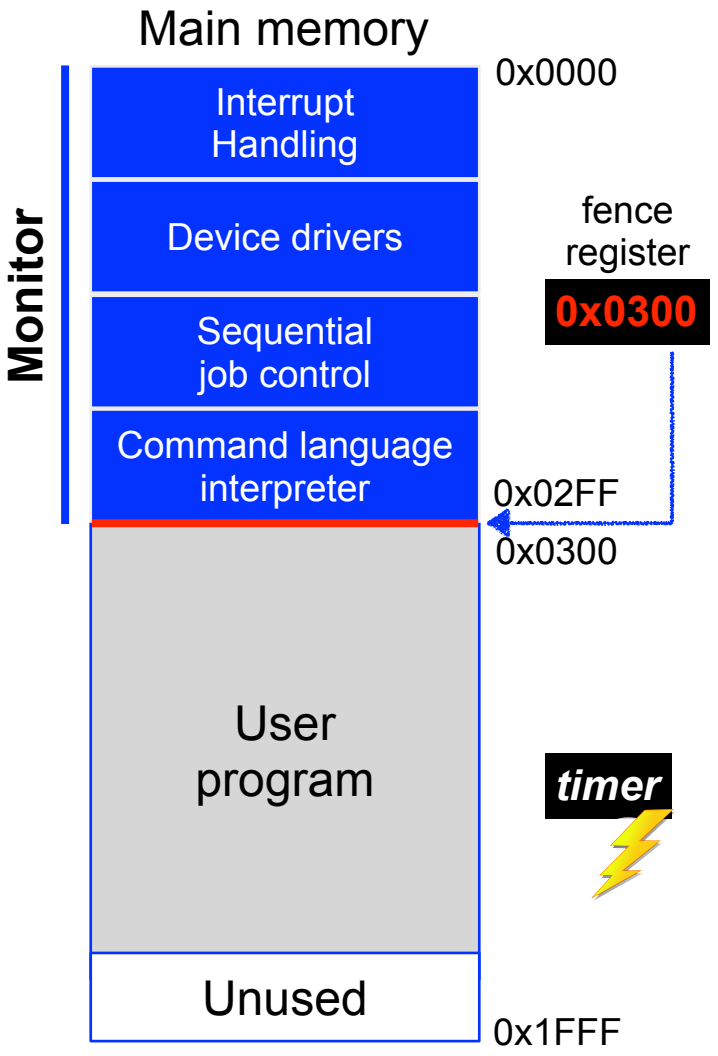
Problems due to erroneous applications:

- Program doesn't terminate,
- writes in the memory of the resident monitor,
- accesses card reader directly and interprets control commands as data

Simple batch systems (from 1955)

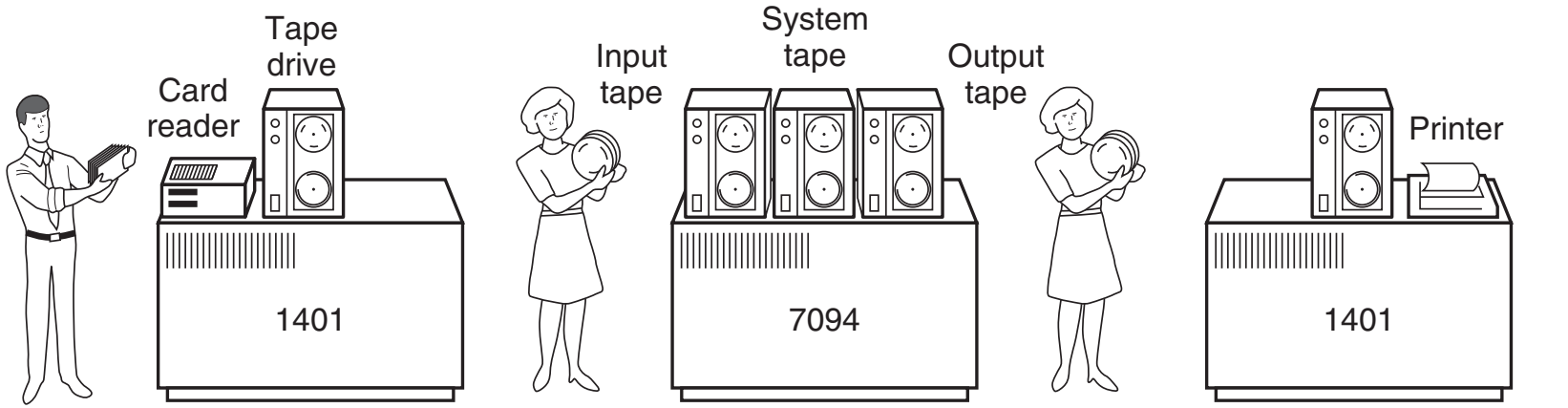
Solutions:

- Addition of a *timer circuit* generating **interrupts**
- **Traps** for erroneous programs
  - *Fence register* to realise primitive **memory protection**
  - **Privileged operating mode** of the CPU (supervisor mode)
    - Deactivates fence register
    - Allows input and output



The Input/Output (I/O) Bottleneck

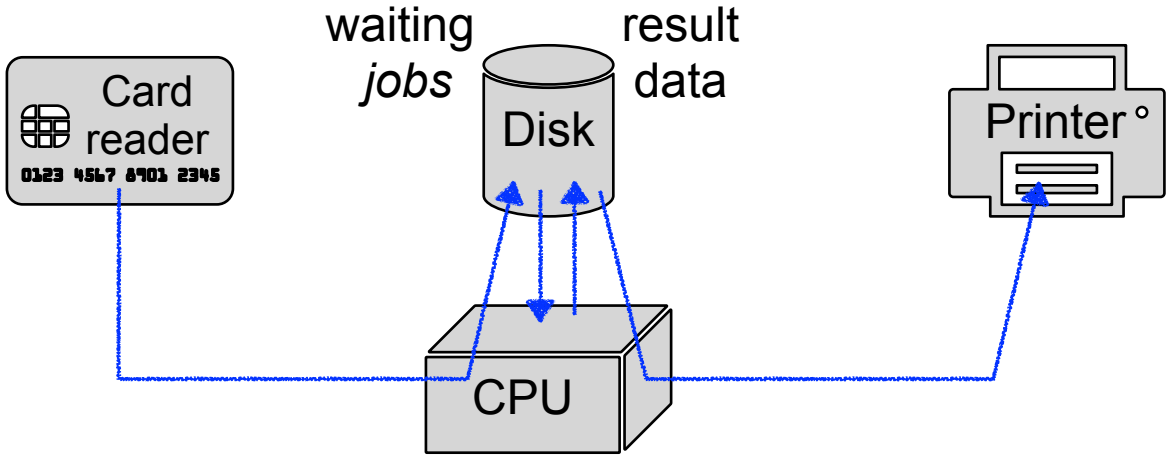
- **Problem:** CPU is faster than card reader and printer
  - valuable compute time is wasted by (active) waiting
- **Solution 1:** ***off line processing***
  - Enabled by magnetic tape drives
  - Parallelisation of I/O using *multiple* satellite computers



[Tanenbaum: Modern operating systems]

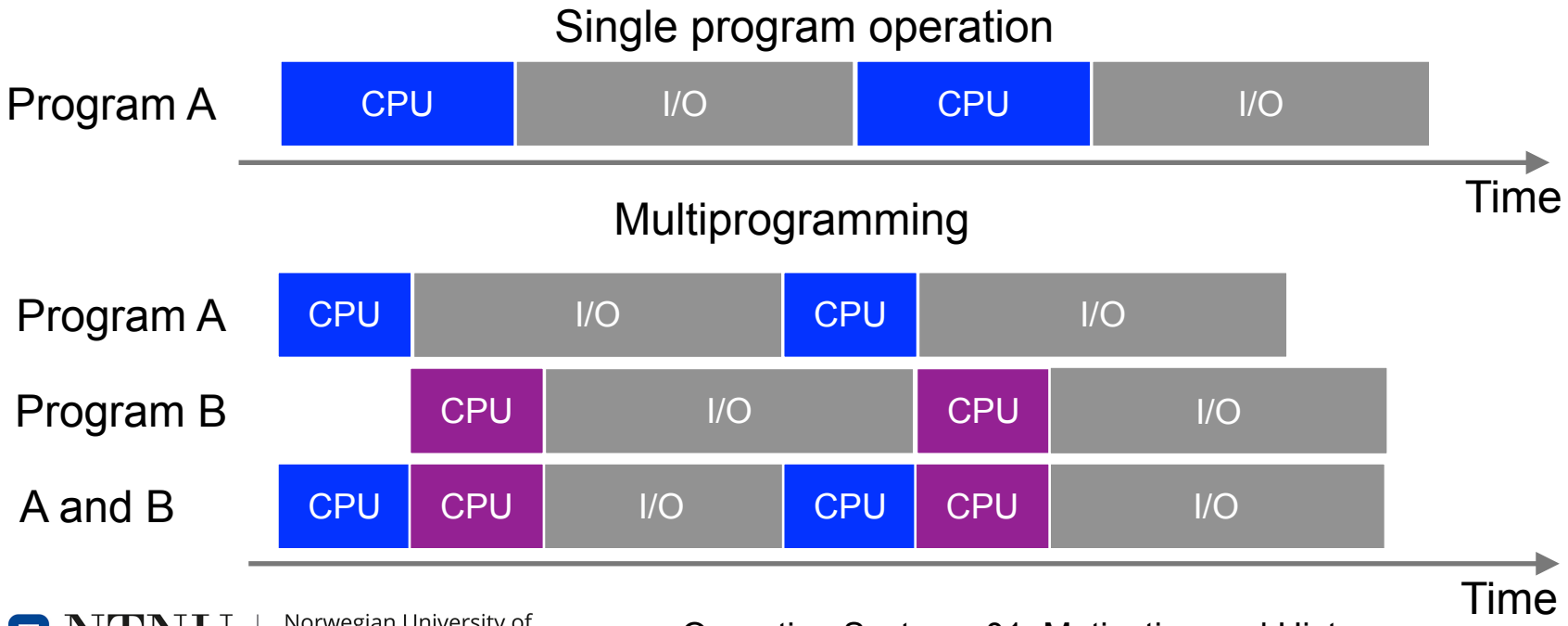
The Input/Output (I/O) Bottleneck

- **Problem:** CPU is faster than card reader and printer
  - valuable compute time is wasted by (active) waiting
- **Solution 2: *spooling***
  - Enabled by magnetic disk drives (random access) and **d**irect **m**emory **a**ccess (DMA)
  - Computation and I/O can now overlap
  - Requires rules for **processor allocation**



Multiprogramming (from 1965)

- Despite *spooling,* a single program does not utilise the CPU efficiently
  - System operation alternates between CPU *bursts* and I/O *bursts,* during with the CPU has to wait
- With *multiprogramming,* the CPU works on multiple jobs at the same time:



Multiprogramming (from 1965)

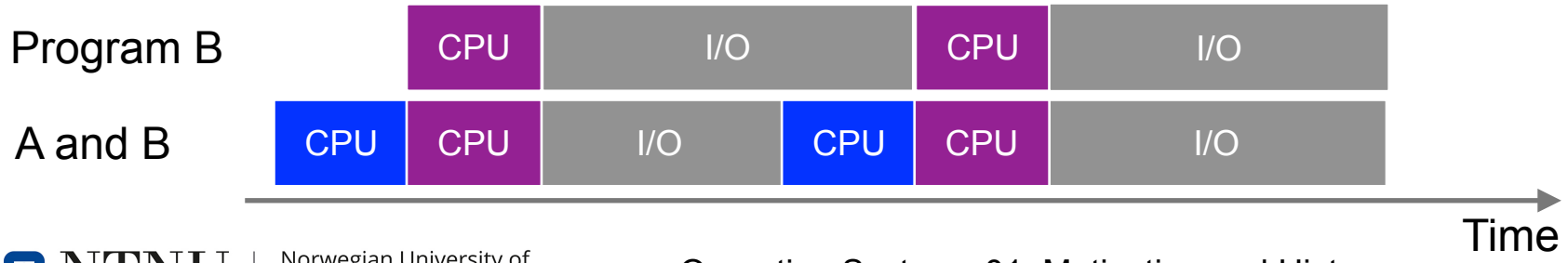
- Despite *spooling,* a single program does not utilise the CPU efficiently
  - System operation alternates between CPU *bursts* and I/O *bursts,* during with the CPU has to wait
- W

The operating system becomes increasingly more complex:

- Handling concurrent I/O activities
- Managing the **main memory** for multiple programs
- Internal management of programs in execution (**processes**)
- Processor **scheduling**
- Multi user operation: **security** and accounting

Pro

Pro



Multiprogramming (from 1965)

Memory management:

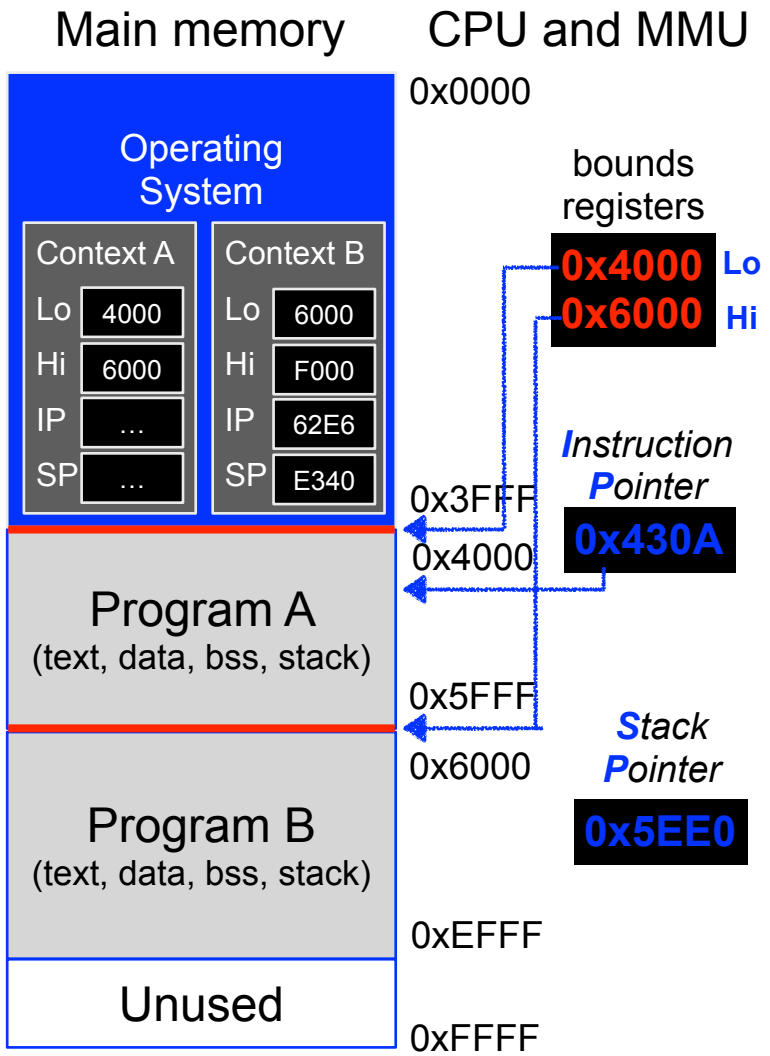
- Programs to be started need an assigned **memory range**

Memory protection:

- Simple fence register is no longer sufficient to isolate processes from each other
- **Solution:** use a simple **memory management unit (MMU)**

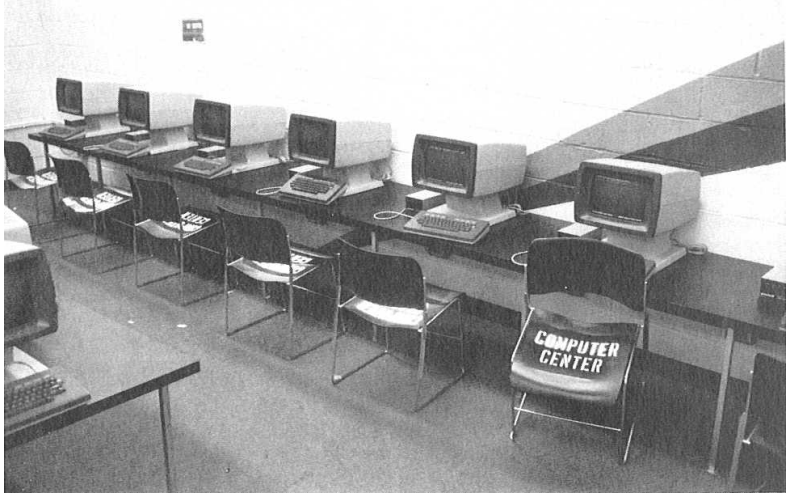
Process management:

- Every “program in execution” has its own **context**
- When switching between processes, the context has to be switched as well



Dialog computing (from 1970)

- New I/O devices enable interactive software
  - Keyboard, screen, later mouse
- Time sharing operation
  - Enables acceptable **response times** for interactive users
  - Timer interrupts ensure the **preemption** of processes which run (too) long
- System programs enable interactive software development
  - Editor, shell, compiler, debugger
- Disks and file systems allow to access programs and data at any time



[Images: Columbia University]

Semester overview

- Review of relevant computer architecture concepts
- Challenges and tasks of operating systems
- Control flow abstractions: processes and threads
- Concurrency: mutual exclusion, synchronisation, deadlocks
- Memory management and virtual memory
- Scheduling: uni- and multiprocessor, realtime
- I/O management and disk scheduling
- File management
- Virtual machines and microkernels
- The Cloud, Unikernels and single-address space OS's
- Embedded systems and non-functional properties
- Operating system security

References

1. Ola Nordal, “Tool or Science? The History of Computing at the Norwegian University of Science and Technology”, IFIP WG9.7 First Working Conference on the History of Nordic Computing (HiNC1), June 16-18, 2003, Trondheim, Norway. Springer. p. 26. ISBN 9780387241685.
2. NUSSE at the Norsk tekniske museum. http://www.nusse.org
3. Erik Høg, “GIER: A Danish computer from 1961 with a role in the modern revolution of astronomy”, in Gudrun Wolfschmidt: Vom Abakus zum Computer - Begleitbuch zur Ausstellung "Geschichte der Rechentechnik", 2015-2019
4. Knut Skog (2002). “Da NTH fikk sin kollektive PC: GIER”, http://datamuseum.dk/w/images/1/1f/ArtikkelGIER-NTH.pdf
5. GIER Simulator. https://ddhf.dk/site_dk/rc/giersimulator/tutorial/tutorial.shtml
6. The first 50 years – A history of digital computing at NTNU https://www.youtube.com/watch?v=wFIyw9zIhrw